Episodic Memory Reader: Learning What to Remember for Question Answering from Streaming Data

Moonsu Han[1*] Minki Kang[1*] Hyunwoo Jung[1] Sung Ju Hwang[1,2]

KAIST[1], Daejeon, South Korea
AITRICS[2], Seoul, South Korea
{mshan92, zzxc1133, hyunwooj, sjhwang82}@kaist.ac.kr

Abstract

We consider a novel question answering (QA) task where the machine needs to read from large streaming data (long documents or videos) without knowing when the questions will be given, which is difficult to solve with existing QA methods due to their lack of scalability. To tackle this problem, we propose a novel end-to-end deep network model for reading comprehension, which we refer to as **E**pisodic **M**emory **R**eader (EMR) that sequentially reads the input contexts into an external memory, while replacing memories that are less important for answering *unseen* questions. Specifically, we train an RL agent to replace a memory entry when the memory is full, in order to maximize its QA accuracy at a future timepoint, while encoding the external memory using either the GRU or the Transformer architecture to learn representations that considers relative importance between the memory entries. We validate our model on a synthetic dataset (bAbI) as well as real-world large-scale textual QA (TriviaQA) and video QA (TVQA) datasets, on which it achieves significant improvements over rule-based memory scheduling policies or an RL-based baseline that independently learns the query-specific importance of each memory.

1 Introduction

Question answering (QA) problem is one of the most important challenges in Natural Language Understanding (NLU). In recent years, there has been drastic progress on the topic, owing to the success of deep learning based QA models (Sukhbaatar et al., 2015; Seo et al., 2016; Xiong et al., 2016; Hu et al., 2018; Back et al., 2018; Devlin et al., 2018). On certain tasks such as machine reading comprehension (MRC), where the problem is to find the span of the answer within a given paragraph (Rajpurkar et al., 2016), the deep-learning based QA models have even surpassed human-level performances.

Despite such impressive achievements, it is still challenging to model question answering with document-level context (Joshi et al., 2017), where the context may include a long document with a large number of paragraphs, due to problems such as difficulty in modeling long-term dependency and computational cost. To overcome such scalability problems, researchers have proposed pipelining or confidence based selection methods that combine paragraph-level models to obtain a document-level model (Joshi et al., 2017; Chen et al., 2017; Clark and Gardner, 2018; Wang et al., 2018b). Yet, such models are applicable only when questions are given beforehand and all sentences in the document can be stored in memory.

However, in realistic settings, the amount of context may be too large to fit into the system memory. We may consider query-based context selection methods such as ones proposed in Indurthi et al. (2018) and Min et al. (2018), but in many cases, the question may not be given when reading in the context, and thus it would be difficult to select out the context based on the question. For example, a conversation agent may need to answer a question after numerous conversations in a long-term time period, and a video QA model may need to watch an entire movie, or a sports game, or days of streaming videos from security cameras before answering a question. In such cases, existing QA models will fail to solve the problem due to memory limitation.

In this paper, we target a novel problem of solving question answering problem with streaming data as context, where the size of the context could be significantly larger than what the memory can accommodate (See Figure 1). In such a case, the

[*] Equal contribution

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4407–4417
Florence, Italy, July 28 - August 2, 2019.

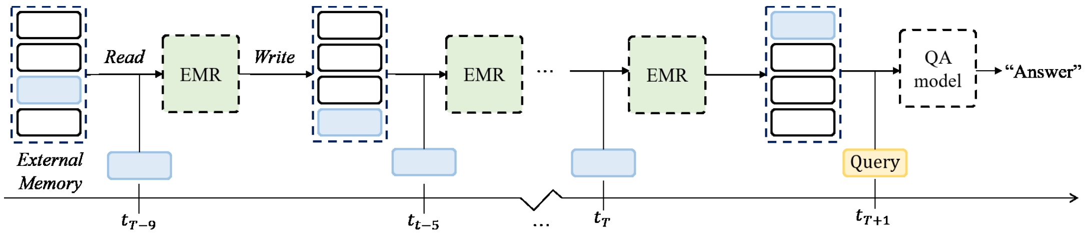


Figure 1: **Concept:** We consider a novel problem of learning from streaming data, where the QA model may need to answer a question that is given after reading in unlimited amount of context. To solve this problem, our Episodic Memory Reader (EMR) learns to retain the most important context vectors in an external memory, while replacing the memory entries in order to maximize its accuracy on an unseen question given at a future timestep.

model needs to carefully manage what to remember from this streaming data such that the memory contains the most informative context instances in order to answer an unseen question in the future. We pose this memory management problem as a learning problem and train both the memory representation and the scheduling agent using reinforcement learning.

Specifically, we propose to train the memory module itself using reinforcement learning to replace the most uninformative memory entry in order to maximize its reward on a given task. However, this is a seemingly ill-posed problem since for most of the time, the scheduling should be performed without knowing which question will arrive next. To tackle this challenge, we implement the policy network and the value network that learn not only relation between sentences and query but also relative importance among the sentences in order to maximize its question answering accuracy at a future timepoint. We refer to this network as Episodic Memory Reader (EMR). EMR can perform selective memorization to keep a compact set of important context that will be useful for future tasks in lifelong learning scenarios.

We validate our proposed memory network on a large-scale QA task (TriviaQA) and video question answering task (TVQA) where the context is too large to fit into the external memory, against rule-based and an RL-based scheduling method without consideration of relative importance between memories. The results show that our model significantly outperforms the baselines, due to its ability to preserve the most important pieces of information from the streaming data.

Our contribution is threefold:

- We consider a novel task of learning to remember important instances from streaming data for question answering task, where the size of the memory is significantly smaller than the length of the data stream.
- We propose a novel end-to-end memory-augmented neural architecture for solving QA from streaming data, where we train a scheduling agent via reinforcement learning to store the most important memory entries for solving future QA tasks.
- We validate the efficacy of our model on real-world large-scale text and video QA datasets, on which it obtains significantly improved performances over baseline methods.

2 Related Work

Question-answering There has been a rapid progress in question answering (QA) in recent years, thanks to the advancement in deep learning as well as the availability of large-scale datasets. One of the most popular large-scale QA dataset is Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016) that contains 100K question-answering pairs. Unlike Richardson et al. (2013) and Hermann et al. (2015) that provide multiple-choice QA pairs, SQuAD provides and requires to predict exact locations of the answers. On this span prediction task, attentional models (Pan et al., 2017; Cui et al., 2017; Hu et al., 2018) have achieved impressive performances, with Bi-Directional Attention Flow (BiDAF) (Seo et al., 2016) that uses bi-directional attention mechanism for the context and query being one of the best performing models. TriviaQA (Joshi et al., 2017) is another large-scale QA dataset that includes $950K$ QA pairs. Since the length of each document in Trivia is much longer than SQuAD, with average of $3K$ sentences per document, existing span prediction models (Joshi et al., 2017; Back et al., 2018; Yu et al., 2018) fail to work due to memory limitation, and simply resort to document truncation. Video question answering (Tapaswi et al., 2016; Lei et al., 2018), where video frames are given as context for QA, is another important topic where scalability is an issue. Several models (Kim et al., 2017, 2018; Na et al., 2017; Wang et al., 2018a) propose to solve video QA using attentions and memory augmented networks, to perform composite reasoning over both videos and texts; however, they only focus on short-length videos. Most existing work on QA focus on small-size problems due to memory limitation. Our work, on the other hand, considers a challenging scenario where the context is order of magnitude larger than the memory.

Context selection A few recent models propose to select minimal context from the given document when answering questions for scalability, rather than using the full context. Min et al. (2018) proposed a context selector that generates attentions on the context vectors, in order to achieve scability and robustness against adversarial inputs. Choi et al. (2017) and Indurthi et al. (2018) propose a similar method, but they use REINFORCE (Williams, 1992) instead of linear classifiers. Chen et al. (2017) selects the most relevant documents out of the Wikipedia database with respect to the query using TF-IDF matching, and Wang et al. (2018b) propose to tackle the document ranking problem with RL agents. While these context/document selection methods share our motivation of achieving scability and selecting out the most informative pieces of information to solve the QA task, our problem setting is completely different from theirs since we consider a challenging problem of learning from the streaming data *without knowing* when the question will be given, where the size of the context is much larger than the memory and the question is unseen when training the selection module.

Memory-augmented neural networks Our episodic memory reader is essentially a memory-augmented network (MANN) (Graves et al., 2014; Sukhbaatar et al., 2015; Xiong et al., 2016; Kumar et al., 2016) with a RL-based scheduler. While most existing work on MANN assume that the memory is sufficiently large to hold all the data instances, a few tried to consider memory-scheduling for better scalability. Gülçehre et al. (2016) propose to train an addressing agent using reinforcement learning in order to dynamically decide which memory to overwrite based on the query. This query-specific importance is similar to our motivation, but in our case the query is given after reading in all the context and thus unusable for scheduling, and we perform hard replacement instead of overwriting. Differentiable Neural Computer (DNC) (Graves et al., 2016) extends the NTM to address the issue by introducing a temporal link matrix, replacing the least used memory when the memory is full. However, this method is a rule-based one that cannot maximize the performance on a given task.

3 Learning What to Remember from Streaming Data

We now describe how to solve question answering tasks with streaming data as context. In a more general sense, this is a problem of learning from a long data stream that contains a large portion of unimportant, noisy data (e.g. routine greetings in dialogs, uninformative video frames) with limited memory. The data stream is episodic, where an unlimited amount of data instances may arrive at one time interval and becomes inaccessible afterward. Additionally, we consider that it is not possible for the model to know in advance what tasks (a question in the case of QA problem) will be given at which timestep in the future (See Figure 1 for more details). To solve this problem, the model needs to identify important data instances from the data stream and store them into external memory. Formally, given a data stream (e.g. sentences or images) $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(T)}\}$ where $\boldsymbol{x}^{(t)} \in \mathbb{R}^d$ as input, the model should learn a function $\mathcal{F} : \boldsymbol{X} \mapsto \boldsymbol{M}$ that maps it to the set of memory entries $\boldsymbol{M} = \{\boldsymbol{m}_1, \cdots, \boldsymbol{m}_N\}$ where $\boldsymbol{m}_i \in \mathbb{R}^k$ and $T \gg N$. How can we then learn such a function that maximizes the performance on unseen future tasks without knowing what problems will be given at what time? We formulate this problem as a reinforcement learning problem to train a memory scheduling agent.

3.1 Model Overview

We now describe our model, *Episodic Memory Reader (EMR)* to solve the previously described problem. Our model has three components: (1) an agent $\mathcal{A}$ based on EMR, (2) an external memory $\boldsymbol{M} = [\boldsymbol{m}_1, \cdots, \boldsymbol{m}_N]$, and (3) a solver which solves the given task (e.g. QA) with the external memory. Figure 2 shows the overview of our model. Basically, given a sequence of data instances $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(T)}\}$ that streams through the system, the agent learns to retain the

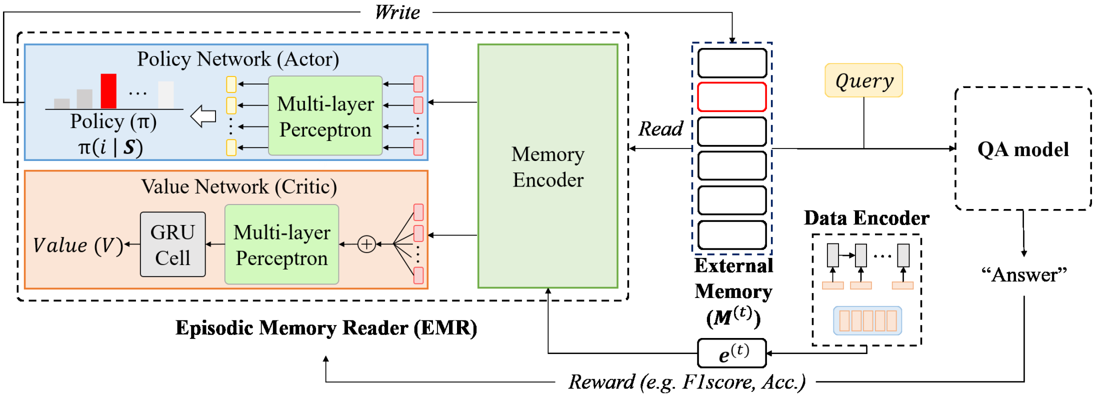


Figure 2: The overview of our Episodic Memory Reader (EMR). EMR learns the policy and the value network to select a memory entry to replace, in order to maximize the reward, defined as the performance on future QA tasks (F1-score, accuracy).

most useful subset in the memory, by interacting with the external memory that encodes the relative importance of each memory entry. When $t \leq N$, the agent simply maps $\boldsymbol{x}^{(t)}$ to $\boldsymbol{m}^{(t)}$. However, when $t > N$, when the memory becomes full, it selects an existing memory entry to delete. Specifically, it outputs an action based on $\pi(i|\boldsymbol{S}^{(t)})$, which denotes the selection of i_{th} memory entry to delete. Here, the state is the concatenation of the memory and the data instance: $\boldsymbol{S}^{(t)} = [\boldsymbol{M}^{(t)}, \boldsymbol{e}^{(t)}]$, where $\boldsymbol{e}^{(t)}$ is the encoded input at timestep t. To maximize the performance on the future QA task, the agent should replace the least important memory entry. When the agent encounters the task $\mathcal{T}$ (QA problem) at timestep $T+1$, it leverages both the memory at timestep T, $\boldsymbol{M}^{(T)}$ and the task information (e.g. question), to solve the task. For each action, the environment (QA module) provides the reward $\mathcal{R}^{(t)}$, that is given either as the F1-score or the accuracy.

3.2 Episodic Memory Reader

Episodic Memory Reader (EMR) is composed of three components: **(1) Data Encoder** that encodes each data instance into memory vector representation, **(2) Memory Encoder** that generates replacement probability for each memory entry, and the **(3) Value Network** that estimates the value of memory as a whole. In some cases, we may use policy gradient methods, in which case the value network becomes unnecessary.

3.2.1 Data Encoder

The data instance $\boldsymbol{x}^{(t)}$ which arrives at time t can be in any data format, and thus we transform it into a k-dimensional memory vector representation $\boldsymbol{e}^{(t)} \in \mathbb{R}^k$ to using an encoder:

$$\boldsymbol{e}^{(t)} = \psi(\boldsymbol{x}^{(t)})$$

where $\psi(\cdot)$ is the data encoder, which could be any neural architecture based on the type of the input data. For example, we could use a RNN if $\boldsymbol{x}^{(t)}$ is composed of sequential data (e.g. a sentence composed of words $\boldsymbol{x}^{(t)} = \{w_1, w_2, w_3, \cdots, w_s\}$) or a CNN if $\boldsymbol{x}^{(t)}$ is an image. After deleting a memory entry $\boldsymbol{m}_i^{(t)}$, we append $\boldsymbol{e}^{(t)}$ at the end of the memory, which then becomes $\boldsymbol{m}_N^{(t+1)}$.

3.2.2 Memory Encoder

Using the memory vector representations $\boldsymbol{M}^{(t)} = [\boldsymbol{m}_1^{(t)}, \cdots, \boldsymbol{m}_N^{(t)}]$ and $\boldsymbol{e}^{(t)}$ generated from the data encoder, the memory encoder outputs a probability for each memory entry by considering its relative importance, and then replaces the most unimportant entry. This component corresponds to the policy network of the actor-critic method. Now we describe our EMR models.

EMR-Independent Since we do not have existing work for our novel problem setting, as a baseline, we first consider a memory encoder that only captures the relative importance of each memory entry independently to the new data instance, which we refer to as EMR-Independent. This scheduling mechanism is adopted from Dynamic Least Recently Use (LRU) addressing introduced in Gülçehre et al. (2016), but different from LRU in that it replaces the memory entry rather than overwriting it, and is trained without query to maximize the performance for unseen future queries. EMR-Independent outputs the importance for each memory entry by comparing them with an embedding of the new data instance $\boldsymbol{x}^{(t)}$ as $\boldsymbol{a}_i^{(t)} = \text{softmax}(\boldsymbol{m}_i^{(t)} \psi(\boldsymbol{x}^{(t)})^T)$. To compute the overall importance of each memory entry, as done in Gülçehre et al. (2016), we compute the exponential moving average as $\boldsymbol{v}_i^{(t)} = 0.1\boldsymbol{v}_i^{(t-1)} + 0.9\boldsymbol{a}_i^{(t)}$. Then, we compute the replacement probabilty of each memory entry with the LRU factor $\gamma^{(t)}$ as follows:

$$
\begin{aligned}
\gamma_i^{(t)} &= \sigma(\boldsymbol{W}_\gamma^T \boldsymbol{m}_i^{(t)} + b_\gamma) \\
\boldsymbol{g}_i^{(t)} &= \boldsymbol{a}_i^{(t)} - \gamma_i^{(t)} \boldsymbol{v}_i^{(t-1)} \\
\pi(i|[\boldsymbol{M}^{(t)}, e^{(t)}]; \theta) &= \text{softmax}(\boldsymbol{g}_i^{(t)})
\end{aligned}
$$

where $i \in [1, N]$ is the memory index, $\boldsymbol{W}_\gamma \in \mathbb{R}^{1\times d}$ and $\boldsymbol{b}_\gamma \in \mathbb{R}$ are the weight matrix and the bias term, $\sigma(\cdot)$ and softmax$(\cdot)$ are sigmoid and softmax functions respectively, and π is the policy of the memory scheduling agent.

EMR-biGRU A major drawback of EMR-Independent is that the evaluation of each memory depends only on the input $\boldsymbol{x}^{(t)}$. In other words, the importance is computed between each memory entry and the new data instance regardless of other entries in the memory. However, this scheme cannot model the relative importance of each memory entry to other memory entries, which is more important in deciding on the least important memory. One way to consider relative relationships between memory entries is to encode them using a bidirectional GRU (biGRU) as follows:

$$
\begin{aligned}
\overrightarrow{\boldsymbol{h}}_i^{(t)} &= GRU_{\theta_{fw}}(\boldsymbol{m}_i^{(t)}, \overrightarrow{\boldsymbol{h}}_{i-1}^{(t)}) \\
\overleftarrow{\boldsymbol{h}}_i^{(t)} &= GRU_{\theta_{bw}}(\boldsymbol{m}_i^{(t)}, \overleftarrow{\boldsymbol{h}}_{i+1}^{(t)}) \\
\boldsymbol{h}_i^{(t)} &= [\overrightarrow{\boldsymbol{h}}_i^{(t)}, \overleftarrow{\boldsymbol{h}}_i^{(t)}] \\
\pi(i|[\boldsymbol{M}^{(t)}, \boldsymbol{e}^{(t)}]; \theta) &= \text{softmax}(MLP(\boldsymbol{h}_i^{(t)}))
\end{aligned}
$$

where $i \in [1, N+1]$ is the memory index, including the index of the encoded input $\boldsymbol{m}_{N+1}^{(t)} = \boldsymbol{e}^{(t)}$, GRU_θ is a Gated Recurrent Unit parameterized by θ, $[\overrightarrow{\boldsymbol{h}}_i^{(t)}, \overleftarrow{\boldsymbol{h}}_i^{(t)}]$ is a concatenation of features. π is the policy of the agent, and MLP is a multi-layer perceptron with three layers with ReLU activation functions. Thus, EMR-biGRU learns the general importance of each memory entry in relation to its neighbors rather than independently computing the importance of each entry with respect to the query, which is useful when selecting out the most important entries among highly similar data instances (e.g. video frames). However, the model may not effectively model long-range relationships between memory entries in far-away slots due to the inherent limitation with RNNs.

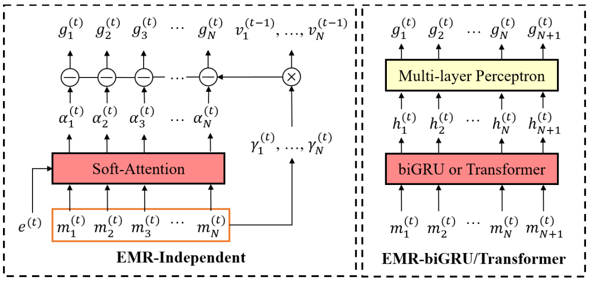


Figure 3: Detailed architecture of memory encoder in EMR-Independent and EMR-biGRU/Transformer.

EMR-Transformer To overcome such suboptimality of RNN-based modeling, we further adopt the self-attention mechanism from Vaswani et al. (2017). With query $\boldsymbol{Q}^{(t)}$, key $\boldsymbol{K}^{(t)}$, and the value $\boldsymbol{V}^{(t)}$ we generate the relative importance of the entries with a linear layer that takes $\boldsymbol{m}^{(t)}$ with the position encoding proposed in Vaswani et al. (2017) as input. With multi-headed attention, each component is projected to a multi-dimensional space; the dimensions for each componenets are $\boldsymbol{Q}^{(t)} \in \mathbb{R}^{H\times N\times \frac{k}{H}}$, $\boldsymbol{K}^{(t)} \in \mathbb{R}^{H\times N\times \frac{k}{H}}$, and $\boldsymbol{V}^{(t)} \in \mathbb{R}^{H\times N\times \frac{k}{H}}$, where N is the size of memory and H is the number of attention heads. Using these, we can formulate the retrieved output using self-attention and memory encoding as follows:

$$
\begin{aligned}
\boldsymbol{A}^{(t)} &= \text{softmax}\left(\frac{\boldsymbol{Q}^{(t)}\boldsymbol{K}^{(t)^T}}{\sqrt{k/H}}\right) \\
\boldsymbol{o}^{(t)} &= \boldsymbol{A}^{(t)}\boldsymbol{V}^{(t)} \\
\boldsymbol{h}^{(t)} &= \boldsymbol{W}_o^T[\boldsymbol{o}_1^{(t)}, \boldsymbol{o}_2^{(t)}, \cdots, \boldsymbol{o}_h^{(t)}] \\
\pi(i|[\boldsymbol{M}^{(t)}, \boldsymbol{e}^{(t)}]; \theta) &= \text{softmax}(MLP(\boldsymbol{h}_i^{(t)}))
\end{aligned}
$$

where i is the memory index, $\boldsymbol{o}_i^{(t)} \in \mathbb{R}^{N\times \frac{d}{h}}$, $[\boldsymbol{o}_1^{(t)}, \boldsymbol{o}_2^{(t)}, \cdots, \boldsymbol{o}_h^{(t)}] \in \mathbb{R}^{N\times k}$ is a concatentation of $\boldsymbol{o}_i^{(t)}$, π is the policy of the agent, and MLP is the same 3-layer multi-layer perceptron used in EMR-biGRU. Memory encoding $\boldsymbol{h}^{(t)}$ is then computed using linear function $\boldsymbol{W}_o \in \mathbb{R}^{d\times d}$ with $\boldsymbol{h}^{(t)}$ as input. Figure 3 illustrates the architecture of the memory encoder for EMR-Independent and EMR-biGRU/Transformer.

3.2.3 Value Network

For solving certain QA problems, we need to consider the future importance of each memory entry.

[Context]

001 World War I (WWI or WW1), also known as the First World War, or the Great War, …

⋮

550 Some war memorials date **the end of the war as being when the Versailles Treaty was signed in 1919**, …

⋮

[Question]

Where was the peace treaty signed that brought World War I to an end?

[Answer]

Versailles castle

Figure 4: Example context and QA pair from TriviaQA.

Especially in textual QA tasks (e.g. TriviaQA), storing the evidence sentences that precede span words may be useful as they may provide useful context. However, using only discrete policy gradient method, we cannot preserve such context instances. To overcome this issue, we use an actor-critic RL method (A3C) (Mnih et al., 2016) to estimate the sum of future rewards at each state using the value network. The difference between the policy and the value is that the value can be estimated differently at each time step and the needs to consider the memory as a whole. To obtain a holistic representation of our memory, we use Deep Sets (Zaheer et al., 2017). Following Zaheer et al. (2017) we sum up all $\boldsymbol{h}_i^{(t)}$ and input them into an MLP (ρ), that consists of two linear layers and a ReLU activation function, to obtain a set representation. Then, we further process the set representation $\rho(\sum_{i=1}^{N} \boldsymbol{h}_i^{(t)})$ by a GRU with the hidden state from the previous time step. Finally, we feed the output of the GRU to a multi-layer perceptron to estimate the value $V^{(t)}$ for the current timestep.

3.3 Training and test

Training Our model learns the memory scheduling policy jointly with the model to solve the task. For training EMR, we choose A3C (Mnih et al., 2016) or REINFORCE (Williams, 1992). At training time, since the tasks are given, we provide the question to the agent at every timestep. At each step, the agent selects the action stochastically from multinomial distribution based on $\pi(i|[\boldsymbol{M}^{(t)}, \boldsymbol{e}^{(t)}]; \theta)$ to explore various states, and make an action. Then, the QA model provides the agent the reward $\mathcal{R}_t$. We use asynchronous multiprocessing method illustrated in Mnih et al. (2016) to train several models at once.

Test At test time, the agent deletes the memory index by following the learned policy π: $\arg\max_i \pi(i|[\boldsymbol{M}^{(t)}, \boldsymbol{e}^{(t)}]; \theta)$. Contrarily from the training step, the model observes the question only at the end of the data stream. When encountering the question, the model solves the task using the data instances kept in the external memory.

4 Experiment

We experiment our EMR-biGRU and EMR-Transformer against several baselines:

1) FIFO (First-In First-Out). A rule-based memory scheduling policy that replaces the oldest memory entry.

2) Uniform. A policy that replaces all memory entries with equal probability at each time.

3) LIFO (Last-In First-Out). A policy that replaces the newest data instance. That is, it first fills in the memory and then discards all following data instances.

4) EMR-Independent. A baseline EMR which learns the importance of each memory entry only relative to the new data instance.

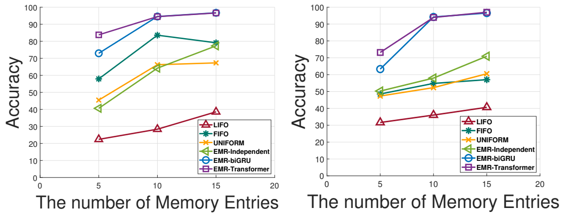
5) EMR-biGRU. An EMR implemented using a biGRU, that considers relative importance of each memory entry to its neighbors when learning the memory replacement policy.

6) EMR-Transformer. An EMR that utilizes Transformer to model the global relative importance between memory entries.

The codes for the baseline models and our models are available at `https://github.com/h19920918/emr`. In the next subsections, we present experimental results on bAbI, TriviaQA, and TVQA datasets. For more experimental results, please see **supplementary file**.

4.1 bAbI

Dataset bAbI (Weston et al., 2015) dataset, which is a synthetic dataset for episodic question answering, consists of 20 tasks with small amount of vocabulary, that can be solved by remembering a person or an object. Among the 20 tasks, we select Task 2, which requires to remember two supporting facts, to evaluate our model. Additionally, we generate noisy version of this task using the open-source template provied by Weston et al. (2015). Each episode of both tasks contains five questions, where all questions share the same context sentences. For Noisy task, we inject noise sentences that has nothing to do with the given task, to validate the effitiveness of our model. In this dataset, 60% of the episodes have no noise sentence, 10% have approximately 30% noise sentences, 10% have approximately 45% noise sen-



(a) Original

(b) Noisy

Figure 5: QA performance of baseline models and EMR-variants. The reported results are averages over 3 runs.

tences and 10% of dataset have approximately 60% noise sentences. The length of each episode is fixed to 45 (5 questions + 40 facts), and a question appears after the arrival of 8 factual sentences.

Experiment Details We adopted MemN2N (Sukhbaatar et al., 2015) for this experiment, which consists of an embedded layer and a multi-hop mechanism that extracts high-level inference. We use MemN2N with position encoding representation, 3 hops and adjacent weight tying. We set the dimension of memory representations to $k = 20$ and compare our model and the baselines on the Original and Noisy tasks. To generate the memory representation $\boldsymbol{m}_i$, we use the sum of the three hop value memories from the base MemN2N. We experiment with varying memory size: $5, 10$ and 15. We train our model and the baselines using ADAM optimizer (Kingma and Ba, 2014) with the learning rate 0.0005 for 400K steps on both tasks.

Results and Analysis In Figure 5, we report the experiment results for Original and Noisy tasks. Both our model (EMR-biGRU and EMR-Transformer) outperform the baselines, especially with higher gain in the case of Noisy dataset. EMR-independent, which does not consider relative importance among the memory entries, performs worse or simlar to FIFO baseline. The results suggest that our methods are able to retain the supporting facts even with small number of memory entries. For further analysis for the experiments on the bAbI dataset, please **see supplementary file**.

4.2 TriviaQA

Dataset TriviaQA (Joshi et al., 2017) is a realistic text-based question answering dataset which includes 950K question-answer pairs from 662K documents collected from Wikipedia and the web. This dataset is more challenging than standard QA benchmark datasets such as Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016), as the answers for a question may not be directly obtained by span prediction and the context is very long (Figure 4). Since conventional QA models (Seo et al., 2016; Back et al., 2018; Yu et al., 2018; Devlin et al., 2018) are span prediction models, on TriviaQA they only train on QA pairs whose answers can be found in the given context. In such a setting, TriviaQA becomes highly biased, where the answers are mostly spanned in the earlier part of the document (Figure 6). We evaluate our work only on the Wikipedia domain since most previous work report similar results on both domains. While TriviaQA dataset consists of both human-verified and machine-generated QA subsets, we use the human-verified subset only since the machine-generated QA pairs are unreliable. We use the validation set for test since the test set does not contain labels.

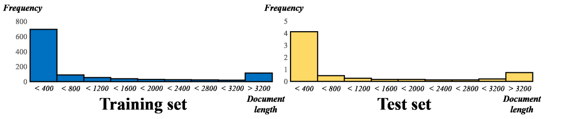
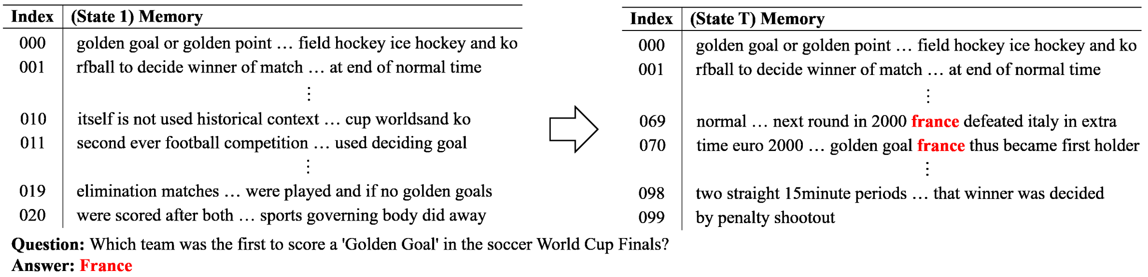


Figure 6: The histogram of number of answers for each document length for TriviaQA dataset.

Experiment Details We employ the pre-trained model from Deep Bidirectional Transformers (BERT) (Devlin et al., 2018), which is the current state-of-the-art model for SQuAD challenge, that trains several Transformers in Vaswani et al. (2017) for pretraining tasks for predicting the indices of the exact location of an answer. We embed 20 words into each memory cell using a GRU and set the number of cells to 20, thus the memory can hold 400 words at maximum. This is a reasonable restriction since BERT limits the maximum number of word tokens to 512, including both the context and the query.

Results and Analysis We report the performance of our model on the TriviaQA using both ExactMatch and F1-score in Table 1. We see that EMR models which consider the relative importance between the memory entries (EMR-biGRU and EMR-Transformer) outperform both the rule-based baselines and EMR-Independent. One interesting observation is that LIFO performs quite well unlike the other rule-based scheduling poli-



Index	(State 1) Memory
000	golden goal or golden point … field hockey ice hockey and ko
001	rfball to decide winner of match … at end of normal time
	⋮
010	itself is not used historical context … cup worldsand ko
011	second ever football competition … used deciding goal
	⋮
019	elimination matches … were played and if no golden goals
020	were scored after both … sports governing body did away

Index	(State T) Memory
000	golden goal or golden point … field hockey ice hockey and ko
001	rfball to decide winner of match … at end of normal time
	⋮
069	normal … next round in 2000 **france** defeated italy in extra
070	time euro 2000 … golden goal **france** thus became first holder
	⋮
098	two straight 15minute periods … that winner was decided
099	by penalty shootout

Question: Which team was the first to score a 'Golden Goal' in the soccer World Cup Finals?
Answer: **France**

Figure 7: An example of how our model operates in order to solve the memory retention problem. Episodic Memory Reader (EMR) sequentially reads the sentences one by one while replacing least important memories. State 1 and State T represent memory entries in the in the initial state and the last state, respectively. Our EMR retrains the sentences which contain the word 'France' (in bold red fonts) in order to answer the given question.

Table 1: Q&A accuracy on the TriviaQA dataset. model.

Model	ExactMatch	F1
FIFO	24.53	27.22
Uniform	28.30	34.39
LIFO	46.23	50.10
EMR-Independent	38.05	41.15
EMR-biGRU	**52.20**	**57.57**
EMR-Transformer	48.43	53.81

cies, but this is due to the dataset bias (See Figure 6) where most answers are spanned in earlier part of the documents. To further see whether this improvement is from its ability to remember important context, we examine the sentences that remain in the memory after EMR finishes reading all the sentences in Figure 7. We see that EMR remembered the sentences that contain key words that is required to answer the future question.

4.3 TVQA

Dataset TVQA (Lei et al., 2018) is a localized, compositional video question-answering dataset that contains 153K question-answer pairs from 22K clips spanning over 460 hours of video. The questions are multiple choice questions on the video contents, where the task is to find a single correct answer out of five candidate answers. The questions can be answered by examining the annotated clip segments, which spans around 30 frames per clip (See Figure 8). The average number of frames for each clip is 229. In addition to the video frames, the dataset also provides subtitles for each video frame. Thus solving the questions requires compositional reasoning capability over both a large number of images and texts.

Experiment Details As for the QA module, we use Multi-stream model for Multi-Modal Video QA, which is the attention-based baseline model provided in (Lei et al., 2018). For efficient training, we use features extracted from a ResNet-101 pretrained on the ImageNet dataset. For embedding subtitles and question-answering pairs, we use GloVe (Pennington et al., 2014). For training, we restrict the number of memory entries for our episodic reader as 20, where each memory entry contains the encoding of a video frame and the subtitle associated with the frame, where the former is encoded using CNN and the latter using GRU. We train our model and the baseline models using the ADAM optimizer (Kingma and Ba, 2014), with the initial learning rate of 0.0001. Unlike from the experiments on TriviaQA, we use REINFORCE (Williams, 1992) to train the policy. This is because TVQA is composed of consecutive image frames captured within a short time interval, which tend to contain redundant information. Thus the value network of the actor-critic model fails to estimate good value of the given state since deleting a good frame will not result in the loss of QA accuracy. Thus we compute the reward $\mathcal{R}^{(t)}$ as the accuracy difference between at time step t and $t-1$ then use only the policy with non-episodic REINFORCE for training. With this method, if the task fails to solve the question after deleting certain frame, the frame is considered as important, and unimportant otherwise.

Results and Analysis We report the accuracy on TVQA as a function of memory size in Figure 9. We observe that EMR variants significantly outperform all baselines, including EMR-Independent. We also observe that the models perform well even when the size of the memory is increased to as large as 60, which was never encountered during the training stage where the number of memory entries was fixed as 20. When the size of memory is small, the gap between different models are larger, with EMR-Transformer obtain-

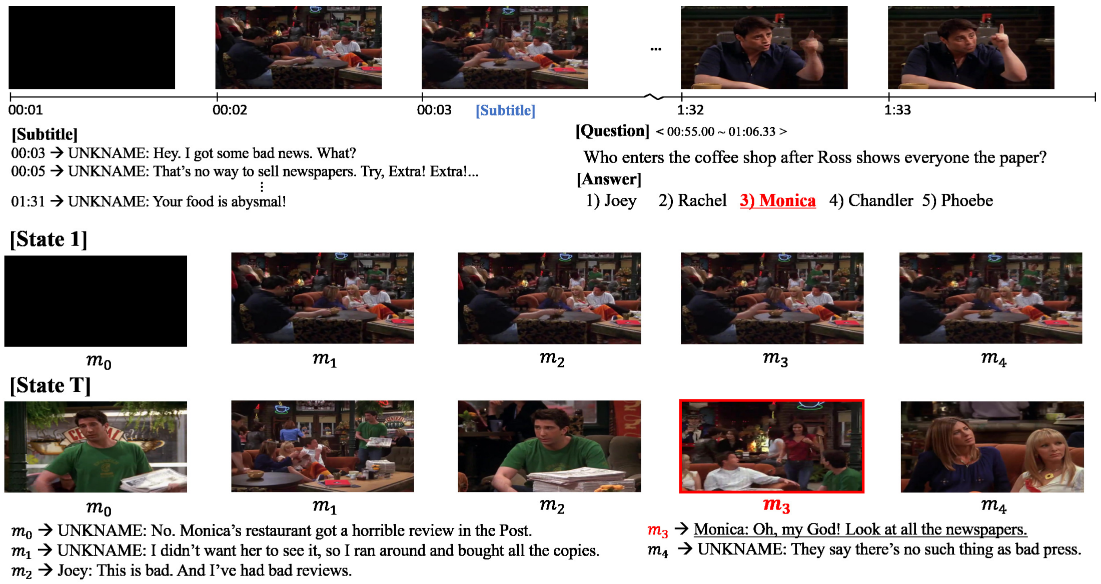

00:01
00:02
00:03
[Subtitle]
1:32
1:33
[Subtitle]
00:03 → UNKNAME: Hey. I got some bad news. What?
00:05 → UNKNAME: That's no way to sell newspapers. Try, Extra! Extra!...
01:31 → UNKNAME: Your food is abysmal!
[Question] < 00:55.00 ~ 01:06.33 >
Who enters the coffee shop after Ross shows everyone the paper?
[Answer]
1) Joey 2) Rachel 3) Monica 4) Chandler 5) Phoebe
[State 1]
m_0
m_1
m_2
m_3
m_4
[State T]
m_0
m_1
m_2
m_3
m_4
m_0 → UNKNAME: No. Monica's restaurant got a horrible review in the Post.
m_1 → UNKNAME: I didn't want her to see it, so I ran around and bought all the copies.
m_2 → Joey: This is bad. And I've had bad reviews.
m_3 → Monica: Oh, my God! Look at all the newspapers.
m_4 → UNKNAME: They say there's no such thing as bad press.

Figure 8: An example of TVQA dataset and a visualization of how our model operates. The answer in red is the ground truth and the answer with underline is the predicted answer from our QA model. At state 1, since the memory is empty, our model encodes every video frame into the memory until the memory becomes full. At state T, after reading in all the video frames, it retains the most informative frames to answer the question. Note that this retention of the important frames is done without knowing the question in advance.

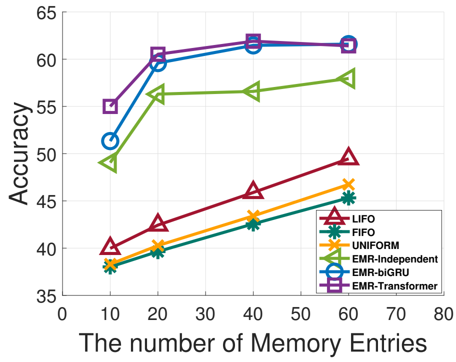


Figure 9: QA accuracy of various memory scheduling policies on the TVQA dataset, reported as a function of the number of memory entries.

ing the best accuracy, which may be due to its ability to capture global relative importance of each memory entry. However, the gap between EMR-Transformer and EMR-biGRU diminishes as the size of memory increases, since then the size of the memory becomes large enough to contain all the frames necessary to answer the question.

As qualitative analysis, we further examine which frames and subtitles were preserved in the external memory after the model has read through the entire sequence in Figure 8. To answer the question for this example, the model should consider the relationship between two frames, where the first frame describes Ross showing the paper to others, and the second frame describes Monica entering the coffee shop. We see that our model kept both frames, although it did not know what the question will be.

5 Conclusion

We proposed a novel problem of question answering from streaming data, where the model needs to answer a question that is given after reading through unlimited amount of context (e.g. documents, videos) that cannot fit into the system memory. To handle this problem, we proposed *Episodic Memory Reader (EMR)*, which is basically a memory-augmented network with RL-based memory-scheduler, that learns the relative importance among memory entries and replaces the entries with the lowest importance to maximize the QA performance for future tasks. We validated EMR on three QA datasets against rule-based memory scheduling as well as an RL-baseline that does not model relative importances among memory entries, which it significantly outperforms. Further qualitative analysis of memory contents after learning confirms that such good performance comes from its ability to retain important instances for future QA tasks.

Acknowledgements

This work was supported by Clova, NAVER Corp. and Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2016-0-00563, Research on Adaptive Machine Learning Technology Development for Intelligent Autonomous Digital Companion).

References

Seohyun Back, Seunghak Yu, Sathish Reddy Indurthi, Jihie Kim, and Jaegul Choo. 2018. Memoreader: Large-scale reading comprehension through neural memory controller. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*.

Eunsol Choi, Daniel Hewlett, Jakob Uszkoreit, Illia Polosukhin, Alexandre Lacoste, and Jonathan Berant. 2017. Coarse-to-fine question answering for long documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*.

Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. 2017. Attention-over-attention neural networks for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *CoRR*, abs/1410.5401.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwinska, Sergio Gomez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. 2016. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476.

Çaglar Gülçehre, Sarath Chandar, Kyunghyun Cho, and Yoshua Bengio. 2016. Dynamic neural turing machine with soft and hard addressing schemes. *CoRR*, abs/1607.00036.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*.

Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and Ming Zhou. 2018. Reinforced mnemonic reader for machine reading comprehension. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*.

Sathish Reddy Indurthi, Seunghak Yu, Seohyun Back, and Heriberto Cuayáhuitl. 2018. Cut to the chase: A context zoom-in network for reading comprehension. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*.

Kyung-Min Kim, Seong-Ho Choi, Jin-Hwa Kim, and Byoung-Tak Zhang. 2018. Multimodal dual attention memory for video story question answering. In *Computer Vision - ECCV 2018*.

Kyung-Min Kim, Min-Oh Heo, Seong-Ho Choi, and Byoung-Tak Zhang. 2017. Deepstory: Video story QA by deep embedded memory networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016*.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L. Berg. 2018. TVQA: localized, compositional video question answering. *CoRR*, abs/1809.01696.

Sewon Min, Victor Zhong, Richard Socher, and Caiming Xiong. 2018. Efficient and robust question answering from minimal context over documents. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016*.

Seil Na, Sangho Lee, Jisung Kim, and Gunhee Kim. 2017. A read-write memory network for movie story understanding. In *IEEE International Conference on Computer Vision, ICCV 2017*.

Boyuan Pan, Hao Li, Zhou Zhao, Bin Cao, Deng Cai, and Xiaofei He. 2017. MEMEN: multi-layer embedding with memory networks for machine comprehension. *CoRR*, abs/1707.09098.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016*.

Matthew Richardson, Christopher J. C. Burges, and Erin Renshaw. 2013. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013*.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*.

Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2016. Movieqa: Understanding stories in movies through question-answering. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*.

Bo Wang, Youjiang Xu, Yahong Han, and Richang Hong. 2018a. Movie question answering: Remembering the textual cues for layered visual contents. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018b. R$\hat{3}$: Reinforced ranker-reader for open-domain question answering. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256.

Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016*.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *CoRR*, abs/1804.09541.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan R. Salakhutdinov, and Alexander J. Smola. 2017. Deep sets. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*.